for synchronization of read & write pointers

9 1

First Hit Fwd Refs

Generate Collection Print

L47: Entry 3 of 20 File: USPT Oct 29, 2002

DOCUMENT-IDENTIFIER: US 6473818 B1
**** See image for Certificate of Correction ****
TITLE: Apparatus and method in a network interface device for asynchronously generating SRAM full and empty flags using coded read and write pointer values

Abstract Text (1):
A network interface device includes a random access memory used as a transmit and receive buffer for transmission and reception of data frames between a host computer bus and a packet switched network. The network interface device includes read and write controllers for each of the transmit and receive buffers, where each write controller operates in a clock domain separate from the corresponding read controller. Read and write counters are each implemented as gray code counters that increment a corresponding pointer value by changing a single bit. A synchronization circuit selectively sets a full or empty flag based on an asynchronous comparison of the read and write pointer values. Use of gray code counters for the read pointer value and write pointer value ensures accurate comparisons in a multi-clock environment.

Brief Summary Text (12):
There is also a need for an arrangement enabling the use of a random access memory as a buffer in a network interface device, where potential synchronization problems between the clock domain of the host computer and the clock domain of the network are resolved to enable efficient control of the random access memory during the writing and reading of transmit or receive data.

Brief Summary Text (14):
These and other needs are attained by the present invention, where read and write counters having read and write pointer values and are each configured to change a single bit each time the corresponding pointer value is incremented, and a synchronization circuit selectively sets a full or empty flag based on a comparison of the read and write pointer values.

Brief Summary Text (15):
According to one aspect of the present invention, a method in a network interface device comprises writing data into a random access memory in the network interface device based on a first clock, changing a single bit of a write pointer value in response to each occurrence of the writing step, reading stored data from the random access memory based on a second clock independent from the first clock, changing a single bit of a read pointer value in response to each occurrence of the reading step, comparing the read pointer value and the write pointer value, and selectively setting one of a full and empty flag based on the comparing step. The changing of a single bit in the read and write pointer values ensures that no errors occur due to transitional states during changing of values. Hence, accuracy of full and empty flag values is ensured by eliminating the synchronization problems that normally occur in a multi-clock environment.

Brief Summary Text (16):
Another aspect of the present invention provides a network interface device for storing a data frame, comprising a random access memory, a write controller configured for writing data to the random access memory according to a first clock,

the write controller including a write counter configured for changing a single bit of a write pointer value in response to writing the data into a corresponding memory location in the random access memory, a read controller configured for reading stored data from the random access memory according to a second clock independent from the first clock, the read controller including a read counter configured for changing a single bit of a read pointer value in response to reading the stored data from a corresponding memory location in the random access memory, and a comparison circuit for selectively determining one of a full condition and an empty condition in the random access memory based on the write pointer value and the read pointer value, independent of the first and second clocks. The changing of a single bit by the counters enables the comparison circuit to accurately determine of the full or empty state of the random access memory independent of the different clocks used in the system.

Brief Summary Text (17):
Still another aspect of the present invention provides a network interface device for storing a data frame, comprising a random access memory, a write controller configured for writing the frame into the random access memory according to a first clock, a read controller configured for reading the frame from the random access memory according to a second clock independent from the first clock, a write counter configured for changing a single bit of a write pointer value in response to a received write signal from the write controller, a read counter configured for changing a single bit of a read pointer value in response to a received read signal from the read controller, and a synchronization circuit configured for selectively generating a signal indicating one of a full or empty condition of the random access memory based on the write pointer value and the read pointer value.

Drawing Description Text (3):
FIG. 1 is a block diagram illustrating an exemplary network interface device having a memory controller for writing data frames to and reading data frames from a random access memory according to an embodiment of the present invention.

Detailed Description Text (2):
The present invention will be described with the example of a network interface device in a packet switched network, such as an Ethernet (IEEE 802.3) network. The description will first be given of the network interface device architecture, followed by the arrangement for selectively setting a full or empty flag based on a comparison of the write and read pointer values. It will become apparent, however, that the present invention is also applicable to other network interface device systems, especially frame based data communication systems (e.g., token ring (IEEE 802.5), fiber distributed data interface (FDDI), etc.).

Detailed Description Text (8):
The network interface device 10 also includes a buffer management unit 24 configured for managing DMA transfers via the DMA interface 16b. The buffer management unit 24 manages DMA transfers based on DMA descriptors in host memory that specify start address, length, etc. The buffer management unit 24 initiates a DMA read from system memory into the transmit buffer 18b by issuing an instruction to the DMA interface 16b, which translates the instructions into PCI bus cycles. Hence, the buffer management unit 24 contains descriptor management for DMA transfers, as well as pointers associated with storing and reading data from the memory portion 18. Although the buffer management unit 24 and the memory management unit 22 are shown as discrete components, the two units may be integrated to form a memory management unit controlling all transfers of data to and from the memory unit 18.

Detailed Description Text (20):
The presence of two separate clock domains 56a and 56b in writing and reading to a random access memory 18 requires that the write controller and read controller devices be coordinated and synchronized to ensure that no contention issues arise

due to the relative independence of the two clock domains 56a and 56b. The SRAM MMU 22 includes a synchronization circuit 60 that asynchronously monitors the status of the RX_SRAM 18a and 18b,enabling the memory controllers to read and write to the memory 18 between the two clock domains 56a and 56b. Thus, problems that would ordinarily arise between the two clock domains in the individual memory management units 22a, 22b, 22c and 22d are avoided by use of the synchronization circuit 60 according to a prescribed arbitration logic.

Detailed Description Text (22):
FIG. 4A is a diagram illustrating multiple data frames (F1, F2, etc.) stored in the RX_SRAM 18a. Assume that the RM_MMU 22d is writing a sequence of data frames 64 (frame 1, frame 2, etc.) into RX_SRAM 18a using a write pointer (WP), while the read controller 22c is reading out the data frames from the RX_SRAM 18a to the BIU 16 using a read pointer (RP). The read pointer (RP) value, is increased according to the same sequence used to increment the write pointer (WP) value, enabling use of the memory 18a as a FIFO-type buffer. Although the pointers are disclosed as incremented to adjacent memory locations, other sequencing arrangements (e.g., increment each time by 2, etc.) may be used.

Detailed Description Text (23):
If the read controller discards (e.g., flushes) a transmit data frame and desires to jump to the beginning of the next data frame, the synchronization circuit 60 must be able to track the start and beginning of each data frame to ensure that the read controller 22c properly locates the beginning of the next data frame. As the read and write pointers are incremented to point to the last memory location, they wrap-around to the starting memory location. One embodiment of this wrap-around mechanism involves the use of modulo counters that are relative to the size of the random access memory. An alternative is to simply reset the counters to the starting location value.

Detailed Description Text (24):
According to one embodiment, the synchronization circuit 60 includes read and write pointers for each SRAM 18a and 18b in order to enable the corresponding memory management unit to track the location of stored data. Since the writing and reading operations occur in two independent clock domains 56, however, a condition may arise as shown in FIG. 4B where the read and write pointers are about to point to the same memory location RP1.

Detailed Description Text (25):
For example, assume a read pointer value and a write pointer value are stored in binary counters, where a write pointer has a value (WR=100) and a read pointer in the second independent clock domain transitions from (RD=011) to (RD=100). Since the clock domain 56a and 56b operate independently of each other, a logic comparator performing a comparison between the write pointer and read pointer may erroneously conclude that the read and write pointers have different values at a point in time where the read pointer has a transitional value (e.g., 101, 111, or 000) as the read pointer is being updated. Hence, the attempt to perform an asynchronous comparison between the binary read and write pointers may cause an erroneous conclusion that the read and write pointers are not equal, causing a glitch in the full/empty flag.

Detailed Description Text (26):
One possible solution for preventing asynchronous comparisons during counter transitions is to provide latched outputs for the counter values. However, such an arrangement would severely degrade the timing performance of the random access memory as a buffer device. Notably, this problem affects the generation of the full and empty flag because of the reliance on accurately stored values in the counters. The full and empty flag is set by comparing read and write pointer values, which are stored in a read counter and a write counter, respectively. If the values in the counters are invalid, the full and empty flag will be set incorrectly.

Detailed Description Text (28):
According to the disclosed embodiment, the synchronization circuit 60 asynchronously compares read and write pointer values for each transmit SRAM 18b and receive SRAM 18a, where each counter is configured for changing a single bit of the corresponding pointer value in response to a corresponding signal from the associated MMU controller.

Detailed Description Text (29):
As illustrated in FIG. 5, the disclosed embodiment contemplates the use of write counters 76a, 76d and read counters 78b, 78c external to the synchronization circuit 60. Each of the counters is implemented as gray code counters, thereby necessitating gray code decoders 80a, 80b, 80c, and 80d for writing to and reading from their respective SRAM 18a, 18b. For example, decoder 80a within the XB_MMU receives gray coded pointer values from the write counter 76a. Use of the gray code counter ensures that any asynchronous comparison between the write counter 76a and the read counter 78b does not result in any erroneous values due to multiple bit transitions that may otherwise occur in counters using binary-format representations. The decoder 80a then converts supplied pointer values into binary values that correspond to address locations within the TX_SRAM 18b. The other MMU components, 22b, 22c, and 22d similarly possess decoders for access to the SRAM, operating in a manner described with respect to the XB_MMU.

Detailed Description Text (30):
The read and write pointer values are also processed in the synchronization circuit 60 to set either a full flag or an empty flag. The read pointer value (RD_CTR) from read counter 78b and the write pointer value (WRCTR) from write counter 76a are supplied to the synchronization circuit 60, and the synchronization circuit 60 in response determines the number of bytes in the TX_SRAM 18b by comparing the supplied pointer values. In general, the comparison involves determining the difference between the read pointer value and the write pointer value. The resultant value is compared with the a predetermined value, which is the maximum size of the memory. If the values match, then the full flag is set to one, indicating a full condition. On the other hand, if the difference is zero, an empty condition is determined; thus, the empty flag is set to one.

Detailed Description Text (31):
In the alternative, decoders need not be used if the random access memory locations are themselves gray coded values. That is, the memory addresses are gray coded values in which case the pointer values do not require conversion into binary values. The operations of the synchronization circuit 60, according to the disclosed embodiment, is unaffected by presence or absence of decoders so long as the pointer values are gray codes.

Detailed Description Text (32):
By employing gray codes as read and write pointer values, synchronization issues inherent in a multi-clock environment are resolved.

CLAIMS:

1. A method in a network interface device, the method comprising: receiving data frames based on a first clock domain at a first interface of the network interface device; writing the data frames into a random access memory in the network interface based on the first clock domain; changing a single bit of a write pointer value in response to each occurrence of the writing step; reading stored data from the random access memory based on a second clock domain independent from the first clock domain; changing a single bit of a read pointer value in response to each occurrence of the reading step; forwarding the read data based on the second clock domain to a second interface on the network interface device; comparing the read pointer value and the write pointer value; and selectively setting one of a full

and empty flag based on the comparing step; writing the data to the random access memory based on the binary write address value.

2. The method of claim 1, wherein the read and write pointer values are gray coded pointer values.

3. The method of claim 1, wherein the comparing step comprises determining the difference between the read pointer value and the write pointer value.

4. The method of claim 1, wherein the writing step comprises: decoding the write pointer value to a binary write address value; and writing the data to the random access memory based on the binary write address value.

5. The method of claim 4, wherein the reading step comprises: decoding the read pointer value to a binary read address value; and reading the data from the random access memory based on the binary read address value.

6. The method of claim 1, wherein the step of changing a single bit of the write pointer value comprises changing the single bit to obtain a predetermined write pointer value corresponding to a wrap-around condition in the random access memory.

7. The method of claim 1, wherein the step of changing a single bit of the read pointer value comprises changing the single bit to obtain a predetermined read pointer value corresponding to a wrap-around condition in the random access memory.

8. The method of claim 1, wherein the comparing step includes comparing the read pointer value and the write pointer value independent of the first and second clock domains.

9. A network interface device, comprising: a first interface configured to bi-directionally transmit data, wherein the first interface operates according to a first clock domain; a second interface configured to bi-directionally transmit data, wherein the second interface operates according to a second clock domain; a random access memory; a write controller configured for writing data received from the first interface to the random access memory according to the first clock domain, the write controller including a write counter configured for changing a single bit of a write pointer value in response to writing the data into a corresponding memory location in the random access memory; a read controller configured for reading stored data from the random access memory and outputting the read data to the second interface according to the second clock domain independent from the first clock domain, the read controller including a read counter configured for changing a single bit of a read pointer value in response to reading the stored data from a corresponding memory location in the random access memory; and a comparison circuit for selectively determining one of a full condition and an empty condition in the random access memory based on the write pointer value and the read pointer value, independent of the first and second clock domains.

11. The network interface device of claim 10, wherein the comparison circuit determines the full condition and the empty condition based on a determined difference between the read pointer value and the write pointer value and a predetermined random access memory size.

12. The network interface device of claim 10, further comprising: a write decoder configured for decoding the write pointer value into a binary memory address value; and a read decoder configured for decoding the read pointer value into a binary memory address value.

14. A network interface for passing data frames comprising: a host bus interface,

wherein the host bus interface operates according to a host clock domain; a network media interface, wherein the network media interface operates according to a network clock domain; a first random access memory partition; a second random access memory partition; a first write controller configured for writing data received from the host bus interface to the first random access memory partition according to the host clock domain, the first write controller including a first write counter configured for changing a single bit of a first write pointer value in response t o writ in g the data in to a corresponding memory location in the first random access memory partition; a first read controller configured for reading stored data from the first random access memory partition and outputting the read data to the network media interface according to the network clock domain independent from the host clock domain, the first read controller including a first read counter configured for changing a single bit of a first read pointer value in response to reading the stored data from a corresponding memory location in the first random access memory partition; a comparison circuit configured for selectively determining one of a full condition and an empty condition in the first random access memory partition based on the first write pointer value and the first read pointer value, independent of the host and network clock domains; a second write controller configured for writing data received from the net work media interface to the second random access memory partition according to the network clock domain, the second write controller including a second write counter configured for changing a single bit of a second write pointer value in response to writing the data into a corresponding memory location in the second random access memory partition; a second read controller configured for reading stored data from the second random access memory partition and outputting the read data to the host bus interface according to the host clock domain independent from the network clock domain, the second read controller including a second read counter configured for changing a single bit of a second read pointer value in response to reading the stored data from a corresponding memory location in the second random access memory partition; and the comparison circuit further configured for selectively determining one of a full condition and an empty condition in the second random access memory partition based on the second write pointer value and the second read pointer value, independent of the host and network clock domains.

15. The network interface device of claim 14, further comprising a first decoder and a second decoder for converting the first read pointer value and the first write pointer value into binary values corresponding to memory locations in the first random access memory partition, respectively.

19. The network interface device of claim 14, further comprising a first decoder and a second decoder for converting the second read pointer value and the second write pointer value into binary values corresponding to memory locations in the second random access memory partition, respectively.